Advances in Engineering Software 97 (2016) 85–95

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

A self-tuning system for dam behavior modeling based on evolving artificial neural networks

B. Stojanovic[a,*], M. Milivojevic[b], N. Milivojevic[c], D. Antonijevic[a]

[a] Faculty of Science, University of Kragujevac, Radoja Domanovica 12, 34000 Kragujevac, Serbia
[b] Technical and Business College, Uzice, Trg Sv. Save 34, 31000 Uzice, Serbia
[c] "Jaroslav Cerni" Institute for the Development of Water Resources, 11000 Belgrade, Serbia

ARTICLE INFO

Article history:
Received 3 August 2015
Revised 8 January 2016
Accepted 20 February 2016

Keywords:
Hybrid dam models
Artificial neural networks
Evolving networks
Genetic algorithms
Multiple linear regression
Dam stability

ABSTRACT

Most of the existing methods for dam behavior modeling presuppose temporal immutability of the modeled structure and require a persistent set of input parameters. In real-world applications, permanent structural changes and failures of measuring equipment can lead to a situation in which a selected model becomes unusable. Hence, the development of a system capable to automatically generate the most adequate dam model for a given situation is a necessity. In this paper, we present a self-tuning system for dam behavior modeling based on artificial neural networks (ANN) optimized for given conditions using genetic algorithms (GA). Throughout an evolutionary process, the system performs near real-time adjustment of ANN architecture according to currently active sensors and a present measurement dataset. The model was validated using the Grancarevo dam case study (at the Trebisnjica river located in the Republic of Srpska), where radial displacements of a point inside the dam structure have been modeled as a function of headwater, temperature, and ageing. The performance of the system was compared to the performance of an equivalent hybrid model based on multiple linear regression (MLR) and GA. The results of the analysis have shown that the ANN/GA hybrid can give rather better accuracy compared to the MLR/GA hybrid. On the other hand, the ANN/GA has shown higher computational demands and noticeable sensitivity to the temperature phase offset present at different geographical locations.

1. Introduction

To describe and predict the structural behavior of dams, a number of statistical, deterministic and hybrid mathematical models have been developed over the past decades. Statistical models based on multiple linear regression (MLR) and their advanced forms such as stepwise regression, robust regression, ridge regression and partial least squares regression have been shown to be more or less successful in dam modeling [1–3]. In contrast to statistical modeling, deterministic models require the solving of differential equations, for which closed form solutions could be difficult or impossible to obtain [4]. Therefore, many models that are based on numerical methods, such as the finite element method (FE), have also been developed [5]. Recently, numerical and statistical methods have been enriched with various heuristics from the artificial intelligence (AI) domain, creating hybrid models that combine their advantages.

Some of these artificial intelligence techniques and heuristic algorithms are artificial neural networks (ANN) [6–10], genetic algorithms (GA) [11–13] and particle swarm optimization (PSO). In his paper [8], Mata presented a comparison between MLR and ANN models for the characterization of dam behavior under environmental loads for the Alto Rabagao arch dam. Gholizadeh et al. [10] used a hybrid methodology with a combination of metaheuristics (GA and PSO) and neural networks to propose an efficient soft computing approach to achieve optimal shape design of arch dams that were subjected to natural frequency constraints. Gomes et al. [14] employed PSO for structural truss mass optimization on size and shape, considering frequency constraints. The results showed that the PSO algorithm performed similarly to other methods and even better in some cases. Several recent studies have also described the application of artificial immune algorithm (AIA) techniques, which imitate the function of a natural immune system [15,16]. Xi et al. [15] proposed an immune statistical model to resolve the data analysis problems of dam horizontal crest upstream-downstream displacements.

In a number of papers [17–22] researchers have made a significant effort to find optimal structures of neural networks for various problems. Majdi et al. [17] combined a neural network and genetic

* Corresponding author. Tel.: +381 69 11 543 75.
E-mail addresses: bobi@kg.ac.rs, boban.stojanovic@gmail.com (B. Stojanovic), milovan.milivojevic@vpts.edu.rs (M. Milivojevic), nikola.milivojevic@gmail.com (N. Milivojevic), dantonijevic.kg@gmail.com (D. Antonijevic).

http://dx.doi.org/10.1016/j.advengsoft.2016.02.010
0965-9978/

algorithm for predicting the deformation modulus of rock masses. GA is utilized to find the optimal number of neurons in a hidden layer, and the learning rates and momentum coefficients of hidden and output layers of the network. Using a standard backpropagation gradient descent algorithm, they tested networks with linear and sigmoid activation functions. Zhou et al. [19] presented a combined procedure of the orthogonal design (OD), FE analysis, ANN and GA for inverse modeling of the seepage/leakage problems. The chosen neural network used the sigmoid transfer function and had a fixed number of layers: one input layer, two hidden layers and one output layer. The number of neurons at the hidden layers was determined by minimizing an error function on a test dataset using a trial-and-error method. To obtain a quick training time and high generalization accuracy, the Levenberg–Marquardt backpropagation algorithm (LM) combined with Bayesian regularization is used for training of the network. In the study [18], a hybrid finite element–boundary element analysis (FE–BE) in conjunction with an ANN procedure is proposed for the prediction of dynamic characteristics of an existing concrete gravity dam. The conjugate gradient algorithm (CGA) and the LM algorithm are implemented for fast training of the ANNs. The authors tested neural networks with one hidden layer where the number of neurons was determined by a trial-and-error method. Hooshyaripor et al. [21] showed that a three-layer ANN model is appropriate to deal with a dam breach problem which has two inputs: the height of water behind the dam and the volume of water behind the dam at the failure time, and one output: the peak outflow discharge. In their study a feed-forward neural network model with a single hidden layer is used. Applying Hecht–Nielsen criterion [22], it was found that an ANN with four neurons in the hidden layer has higher performance. The LM algorithm was employed to train the ANN model. As a transfer function, tan-sigmoid and linear functions were employed in the hidden and output layers, respectively.

In our previous work we developed an adaptive system for dam behavior modeling based on a linear regression model optimized for given conditions using genetic algorithms [23]. Throughout the evolutionary process, the system performs near real-time adjustment of regressors in the MLR model according to currently active sensors. Following this idea, we developed a system for dam behavior modeling based on artificial neural networks, capable of adapting to persistent changes in the measuring system and measurement database. In order to achieve a full adaptability of the ANN model in real-world conditions, the system should be able to optimize all significant network parameters according to available input sensors and historical data. To the best of our knowledge, the existing solutions optimize a limited set of ANN parameters only, while the other parameters are chosen arbitrarily, based on experience, literature or trial-error methods. In this paper, we present a novel methodology and a system for automatic generation of an ANN dam model, which optimizes all significant elements of the ANN architecture. Guided by a variant set of input variables, the system permanently optimizes network topology, activation functions and learning algorithms in order to fit the growing measurement database. Optimization of the parameters is performed using genetic algorithms. The quality of the proposed ANN/GA hybrid dam models has been tested on a real-world case study and compared to equivalent dam models based on multiple linear regression and GA (MLR/GA).

2. Theoretical background

2.1. Artificial neural network

The ANN is a simplified mathematical model of a natural neural network. It is a computing system made up of a number of simple, interconnected processing elements or neurons, which process information by a dynamic state response to external inputs [17]. Processing elements are grouped in layers: an input layer, one or more hidden layers and an output layer. The neurons (nodes) are interconnected by weighted links. A special class of ANN are feed-forward networks, which propagate a signal from the input to output layer [24]. A schematic view of a feed-forward network is given in Fig. 1, where **X** denotes a vector of predictor variables, **Y** is a vector of response variables, $\mathbf{w}^{(\mathbf{i},\mathbf{h})}$ is a column-matrix of weighting coefficients between neurons in the input layer and the first hidden layer, and $\mathbf{w}^{(\mathbf{h},\mathbf{o})}$ is a column-matrix of weighting coefficients between neurons in the last hidden layer and the output layer. The term $w_{ij}^{(q-1,q)}$ denotes the weight between ith neuron from $(q-1)$th hidden layer and jth neuron from (q)th hidden layer. In order to improve performance of the neural network, there is an extra neuron assigned to each hidden layer and the output layer, with the role of sending a constant signal x_0 (bias) to all neurons in these layers.

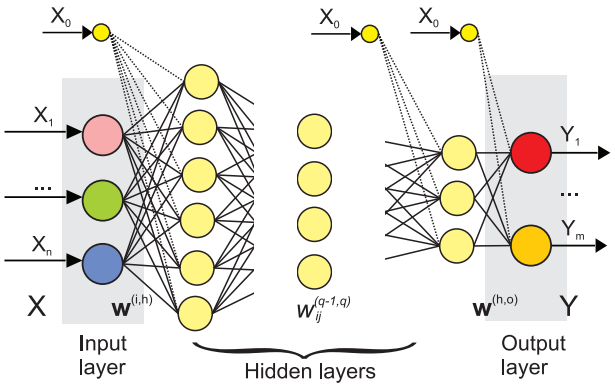

Fig. 1. Structure of feed-forward neural network.

According to the *sigma rule*, the total input $\alpha_j^{(q)}$ into processing element j in qth layer is a weighted sum of all outputs $x_i^{(q-1)}$ from the previous layer. In the same manner, the input signals into neurons of the output layer are calculated as a function of outputs from the last hidden layer. When input signal $\alpha_j^{(q)}$ passes through a neuron, it is processed and transformed to the output signal $x_j^{(q)}$ using an *activation function*. The activation function (AF) is necessary to transform the weighted sum of all signals hitting on a neuron so as to determine its firing intensity [17]. Some of the frequently used activation functions are: *Gaussian*, *log*, *sigmoid*, *bipolar sigmoid*, *sine*, *hyperbolic tangent (TANH)*. Characteristics of various AFs are given in details in [25].

The aim of the learning process is to set weights to the values for which the difference between desired and calculated values (error function) will be minimal. One of the most popular learning algorithms that minimizes error function is a *backpropagation algorithm*. According to the backpropagation algorithm, the error is propagated backward through the network and the weights are adjusted during a number of iterations. The procedure progresses until it reaches convergence of the calculated and expected outputs [17]. There are many variations of the backpropagation learning algorithm. In this work we used some of the best known: The Backpropagation gradient descent algorithm (BPGD) [24] and the Resilient propagation algorithm (RPROP) [26–29].

Design of an ANN is specified by network architecture (such as the number of hidden layers and neurons, type of AFs, etc.) and learning rules. Both the architecture and learning rules are very important, thus good selection of these will give better performance of the network [17]. This task is still an unsolved issue and most researchers use a trial-and-error method to find a

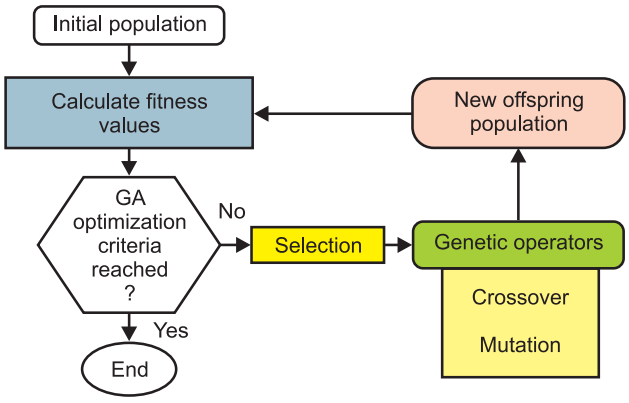

Fig. 2. Schematic view of Genetic Algorithm.

suitable number of hidden layers and nodes [22,30–32]. Different ideas have been stated about the required number of training data. The number of training samples has an influence on the quality of the model obtained. Researchers propose different sizes of training datasets, ranging from 60% to 80% of the available data [33–35]. The learning rules specify an initial set of weights, as well as the learning rate η and the momentum μ in the BPGD algorithm or $\eta^+$and η^- parameters in the RPROP algorithm. The range of initial weights commonly used in literature is $[-0.5, 0.5]$ [36]. In order to reduce the impact of the stochastic nature of initial weights on the quality of the ANN model, Messer and Kittler proposed that the learning process should be repeated at least 5–10 times with different initial weights [17,37].

2.2. Genetic algorithms

Genetic algorithms are search techniques that are inspired by the theory of natural selection, in which strong species have a greater opportunity to survive and pass their genes on to future generations via reproduction [38,39]. GAs are probabilistic algorithms that maintain a population of individuals, $P(t) = x_1(t), \ldots, x_n(t)$, for the iteration (generation) t, where each individual represents a potential solution to a problem. Each solution $x_i(t)$ is evaluated to quantify its fitness. Subsequently, a new population (iteration $t+1$) is formed by selecting better individuals from those of generation t. In GA terminology, a solution vector $\mathbf{x} \in \mathbf{X}$ is called an individual or chromosome and corresponds to a unique solution $\mathbf{x}$ in the solution space. The chromosomes are made of discrete units called genes. Each gene controls one or more features of the chromosome. In this paper, genes are assumed to be binary digits, according to the original implementation of GAs by Holland [38].

GAs use two operators to generate new solutions from existing solutions: crossover and mutation. In crossover, two chromosomes, called parents, are combined together to form new chromosomes, called offspring. The mutation operator introduces random changes into the characteristics of chromosomes, for the purpose of reintroducing genetic diversity back into the population and assisting the search to escape from local optima.

Reproduction involves selecting a set of chromosomes that will survive into the next generation. The procedure of a generic GA is given in Fig. 2.

After the random generation and evaluation of initial solutions, the population is subjected to the iterative process of selection, mating (crossover), mutation and evaluation for the next iteration (generation). The iterative process is terminated when there is a satisfactory quality of solutions or when the maximum number of iterations is reached.

3. The ANN/GA hybrid for dam behavior modeling

3.1. Evolving neural network model using GA

A full adaptability of a real-world dam model requires the ability of the model to deal with the persistent increase of a measurement dataset and a variant set of predictor variables induced by permanent changes and malfunctions in the measuring system. In order to provide an ANN dam model that is fully adaptive to a variant dataset and predictors, in this paper we present a methodology for near real-time optimization of a structural dam model based on artificial neural networks. This concept implies optimization of all elements of the network with the aim to generating a model that best describes structural dam behavior under given conditions. The elements subjected to optimization are the number of hidden layers, the number of neurons per layer, activation functions, learning algorithms and learning rules.

3.1.1. ANN/GA mathematical programming model

Let Ω denote the set of possible ANN models that contain at most $N_{\max}$ hidden layers with a maximum of $n_{\max}$ neurons per layer, where all the neurons use one of the available AFs. In addition, every single ANN model uses one of the offered learning algorithms tuned with the parameters from the recommended ranges.

Let $RMSE_{Test}^{(\psi)}$ denote the root mean squared error that the trained network $ANN^{(\psi)}$ from Ω produces using the testing dataset. Then, a mathematical programming model to find the best ANN topology and the best choice of AF, learning algorithm, and learning rules, can be written in the following form:

Minimize $RMSE_{Test}^{(\psi)}, \quad ANN^{(\psi)} \in \Omega$

Subjected to

$$N^{(\psi)} \leq N_{\max} \tag{1}$$

$$n^{(\psi)} \leq n_{\max} \tag{2}$$

$$AF^{(\psi)} \in \{Gaussian,\ sine,\ TAHN, \ldots\} \tag{3}$$

$$LA^{(\psi)} \in \{BP\ Gradient\ Descent\ , RPROP, \ldots\} \tag{4}$$

$$a \in [a_{\min}^{(\psi)}, a_{\max}^{(\psi)}] \tag{5}$$

where $N^{(\psi)}, n^{(\psi)}, AF^{(\psi)}, LA^{(\psi)}$ are the number of hidden layers, the number of nodes in hidden layers, activation function, and learning algorithm of neural network $ANN^{(\psi)}$, respectively. In Eq. (5), the variable a stands for any of the parameters η, μ, η^+ or η^-. In addition, for the sake of computational efficiency, the same AF is used for all hidden and output neurons.

3.1.2. Genetic algorithm for the neural network optimization

Regarding the complexity of the optimization problem, we used a genetic algorithm as an optimization technique, due to its inherent generality and robustness. The developed optimization methodology (ANN/GA hybrid) is based on the iterative strategy of the GA shown in Fig. 2, where individuals represent neural networks from Ω. Every single individual (neural network $ANN^{(\psi)}$) from the initial population is evaluated in order to determine its fitness. In our case, the networks with lower $RMSE_{Test}^{(\psi)}$ are considered to be better, thus the fitness is calculated as $1/RMSE_{Test}^{(\psi)}$. According to the chosen selection criterion and the elitism concept, networks with the lower fitness are discarded. Through the processes of crossover and mutation, a new generation of neural networks with different topologies, learning algorithms, activation functions, and learning rules is generated. Once the convergence criterion is achieved (defined time, required accuracy...), a population of optimized artificial neural networks is obtained. Since there is only one optimization criterion (RMSE), the network with the best fitness within the final population is chosen as the most accurate one.

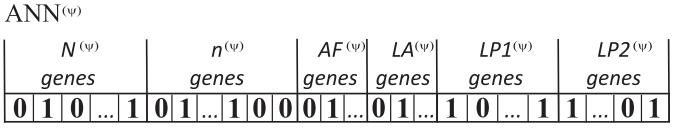

Fig. 3. Genetic structure of an ANN/GA chromosome.

3.1.3. ANN chromosome structure

According to Holland's original approach, individuals in the population pool are in the form of binary chromosomes. The proposed genetic structure of the individuals in a population pool is shown in Fig. 3.

The first two groups of genes represent the number of hidden layers $N^{(\psi)}$ and the number of neurons $n^{(\psi)}$ in these layers. Note that the number of neurons is assumed to be the same in all hidden layers ($n_1^{(\psi)} = \cdots = n_{Nmax}^{(\psi)} = n^{(\psi)}$), in order to keep the chromosome size constant over the population. The next group of genes represents a type of activation function $AF^{(\psi)}$. The $LA^{(\psi)}$ genes in the chromosome define the learning algorithm, while $LP1^{(\psi)}$ and $LP2^{(\psi)}$ genes include the parameters of the learning algorithm (η and μ for the BPGD, or η^+ and η^- for the RPROP algorithm). In order to avoid constraint violations, each of integer variables (including enumerated activation functions and learning algorithms) should be in the range $[1 - 2^l]$, where l is the number of bits in the gene representing that variable. On the other hand, the range for each real variable is set according to recommendations, but its resolution will depend on the number of bits in that gene. Coding variables in this way guarantees that their values will remain inside the prescribed ranges during crossover and mutation processes.

Fig. 4 shows the example of two ANN individuals, where both networks have 3 predictors and one response variable. Binary chromosome $ANN^{(\psi)}$ in Fig. 4a represents an ANN with 2 hidden layers, each with 5 neurons and the TANH activation function. For the sake of brevity, the genes that represent the learning algorithm and learning rules are not shown in the Figure. The $ANN^{(\gamma)}$ chromosome, shown in Fig. 4b, represents a coded ANN with 3 hidden layers, each with 3 neurons and a sinusoidal activation function. Here we used a notation of the form $I/N(n)/O$, where I, N, n and O are the number of predictors, the number of hidden layers, the number of neurons per layer, and the number of response variables, respectively. Accordingly, the topology of $ANN^{(\psi)}$ and $ANN^{(\gamma)}$ individuals can be written in forms *3/2(5)/1* and *3/3(3)/1*.

3.1.4. Genetic algorithm operators

In this study, the single point crossover operator has been used. Once the crossover point is randomly selected, two mating chromosomes are cut at corresponding points, and the sections after the cuts are exchanged (Fig. 5a). To avoid the local minima problem, each bit is independently flipped with a probability of p, using a bit-flip mutation operator (Fig. 5b) [23].

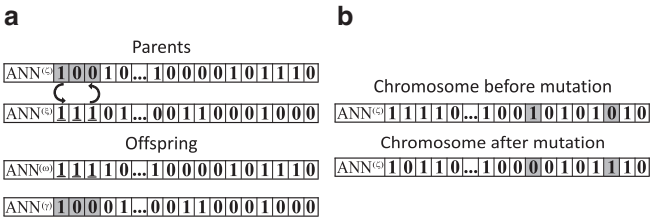

Fig. 5. Crossover (a) and mutation (b) operators.

3.1.5. Evaluation of ANN individuals

Evaluation of each $ANN^{(\psi)}$ individual is realized as shown in Fig. 6. The content of the chromosome defines the activation function, learning algorithm and learning rules used in the neural network represented by that individual. According to the activation function, the data in the learning dataset (LDS) and test dataset (TDS) are normalized. After the generation of random initial values of weighting coefficients w_{ij}, the learning algorithm with defined learning rules is performed as described above.

The learning process is aborted in the overlearning zone [31]. In order to reduce the influence of the stochastic generation of the initial weight matrix ($\mathbf{W}_0^{(\psi,r)}$) on the quality of the $ANN^{(\psi)}$ model, the whole procedure is repeated several times with different initial weights. Finally, the fitness of the $ANN^{(\psi)}$ individual is calculated as $1/\mathrm{RMSE}_{Test}^{(\psi)} = 1/\min(RMSE_{Test}^{(\psi,\mathbf{W}_0^{(\psi,r)})})$, $r = 1, R$, where R is a number of repetitions, arbitrarily chosen by a researcher.

3.1.6. Selection

According to the optimization criterion that is defined in the presented mathematical programming model, all of the individuals (models) are ranked based on $RMSE_{Test}^{(\psi)}$, $\psi \in \Omega$. In this study, we employed binary tournament selection, which performs a tournament between pairs of individuals and selects the winners that pass to the next generation. To assure that the best individuals always survive to the next generation, an elitism strategy has also been used [23].

3.2. Implementation of ANN/GA hybrid for dam behavior modeling

In order to prove the proposed ANN/GA concept, we have developed a self-tuning software system for dam behavior modeling, named DEVONNA (**Dam Evo**lving **N**eural **N**etworks), (Fig. 7).

The structural dam behavior (1) can be considered as a black box that transforms the vector of input variables **X** (2) into the

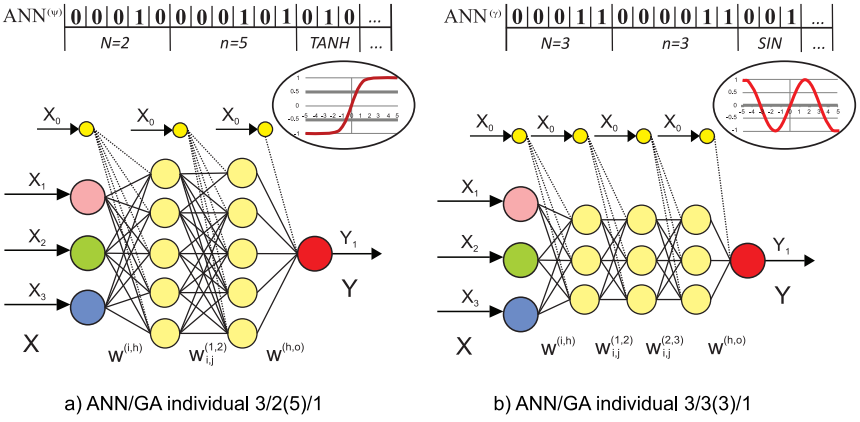

Fig. 4. Examples of the genetic structure of ANN/GA chromosomes.

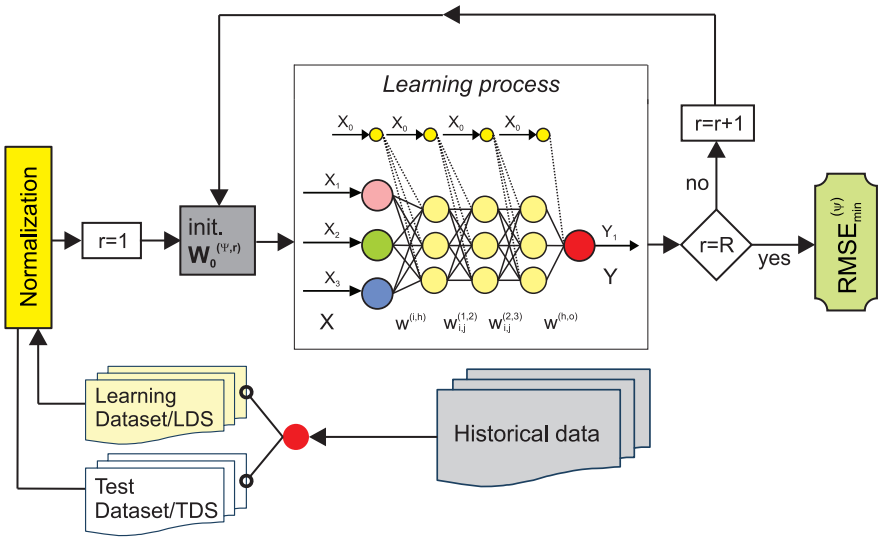

Fig. 6. ANN evaluation process.

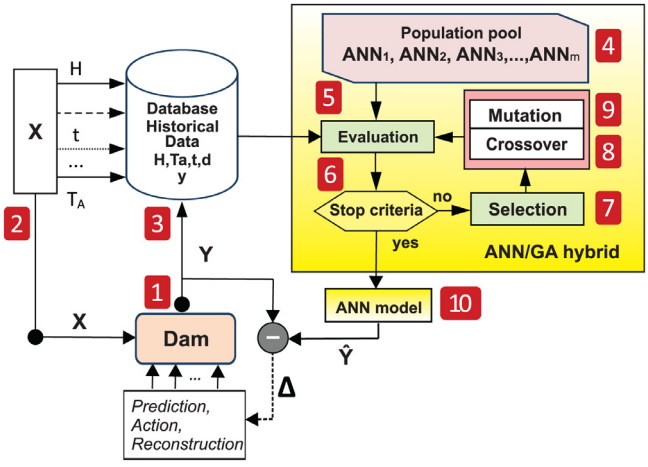

Fig. 7. Schematic view of DEVONNA software solution, a self-tuning system for dam behavior modeling based on evolving neural networks.

vector of output variables **Y**. Measurements of input variables, such as headwater H, air temperature T_a and time t (ageing), as well as the output variables of interest, such as displacements and stresses, are stored in the database (3). Nevertheless, if temperature measurements are incomplete or unavailable, trigonometric functions as temperature cyclic (seasonal) loadings are commonly used in dam modeling to describe deformation patterns as follows [11]:

$$\begin{aligned} &\sin(2\cdot k\cdot \pi\cdot d/365),\ k=1,2,3\ldots \\ &\cos(2\cdot k\cdot \pi\cdot d/365),\ k=1,2,3\ldots \end{aligned} \tag{6}$$

where d is the day of the year.

The ANN/GA hybrid randomly generates an initial population of individuals (binary chromosomes), each representing one ANN (4). In order to find the network that best represents the historical behavior of the dam, the population of the networks is optimized throughout an evolutionary process (5, 6, 7, 8, 9) described in Section 3.1. The optimization procedure is repeated a few times in order to reduce the effects of the inherent stochastic nature of the GA (usually 5 times, [40,41]).

The final product of the DEVONNA system is the optimized mathematical model (10) based on the ANN, which can predict the structural dam behavior $\hat{\mathbf{Y}}$ given the known input variables. This model can be further used for dam behavior prediction, analysis, and comparison with measurements and possible actions.

3.3. Software implementation

The presented DEVONNA system has been developed in a Microsoft .NET software environment and comprises a data acquisition service, an MS SQL database and a module for the optimization of ANN dam models. The special component of the system is the interface to the Encog software library [25], which is employed for the processes of learning the ANN.

4. Validation of the proposed ANN/GA hybrid

The following sections present validation of the proposed ANN/GA hybrid and the developed DEVONNA system using the case study of modeling Grancarevo dam displacements.

4.1. Case study

The Grancarevo dam is the first step of the hydropower system of the Trebisnjica river in the Republic of Srpska. The dam is located 18 km from the river source and 17 km upstream from the city of Trebinje, creating a reservoir of $1278\cdot 10^6$ m^3 (Fig. 8).

The dam body is made of 377000 m^3 of concrete in the form of 31 cantilever blocks, numerated from the right to the left bank (Fig. 9). This dam is a double curvature arch dam (R = 185.48 m), which is 123 m high and has a 439.3 m long crest, with a thickness of 4.6 m at the top and 26.9 m at the bottom of the dam. It is equipped with a monitoring system to measure parameters such as concrete conditions, water and air temperatures, the reservoir water level, horizontal and vertical displacements, rotation, movements of joints, strain, stress, uplift pressure, foundation displacements and seepage, at a total of 465 measuring points.

Three pendulums were installed to measure radial and tangential deformations. In this paper we modeled the radial displacements of point P1 at the dam crest (elevation 403 m.a.s.l.) (Fig. 9a), block 17 (Fig. 9b). Displacements of the point are measured using coordinometer V17-1. In fact, the dam monitoring system usually provides many outputs, which can be used in the assessment of structural behavior. Although artificial neural networks have the ability to model more than one output variable simultaneously, for the sake of clarity, in this work we present the principle using only one output.

The dam was built in 1967 and up to now about 14500 measurements of point P1 displacements have been made. However, there are a lot of missing values in the period between 1967 and

Fig. 8. A view of the Grancarevo dam.

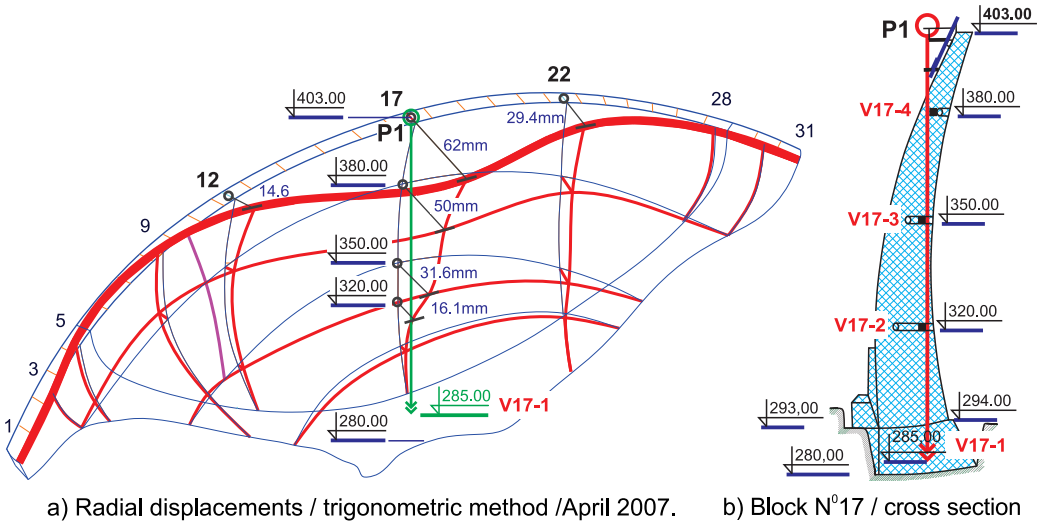

Fig. 9. The radial displacement of point P1 (a) and the cross-section through Block 17 (b).

1984, thus the period from January 1984 to the end of August 2011 was chosen for modeling. For the sake of computational efficiency we used every second data, so the dataset consisted of 5042 data samples in total. According to a *cross-validation* strategy, this dataset is divided into LDS and TDS subsets with a given ratio [42–44].

The effect of hydrostatic pressure on the dam displacements was taken explicitly into account through the headwater H. The range of the input variable H was from 331.05 to 401.28 m. The thermal effect was accounted by the mean daily air temperature T_a(−7.10 to 32.10 °C). Thermal effects are also represented by trigonometric functions as temperature cyclic (seasonal) loadings which can be used to describe deformation patterns through input variable d. The variable d represents the time elapsed from the beginning of the year, ranging from 1 to 365 days. However, at different geographical locations, temperature oscillations can have a phase offset from the beginning of the year. In order to test stability of the models regarding temperature phase offset, we used dummy input variables $d20$ and $d50$ that represent phase offsets of 20 ($d20 = d + 20$) or 50 ($d50 = d + 50$) days, respectively. The ageing effect, as a function of time t, includes the influence of degradation of the material properties during the structural lifetime on measured values. As mentioned before, the radial displacement of point P1, represented by variable y_{17}, is chosen to be the output (response) variable. The range of the displacements in the analyzed period was from +13.26 to +64.4 mm, where radial displacements in downstream direction are considered positive.

An example of the learning dataset, which represents 60% of the available data, is given in Fig. 10. As an effect of hydrostatic pressure, radial displacements are increased with an increase of water level H, and vice versa. In contrast, due to thermal expansion, radial displacements are decreased with a raise of temperature T_a, and vice versa.

4.2. Results

The proposed ANN/GA methodology and the developed DEVONNA system were validated on the data that were measured between 1.1.1984 and 29.08.2011 (D.M.Y date format is used). According to the cross-validation strategy, we performed four series of tests with different LDS:TDS ratios (60:40, 70:30, 75: 25 and 80:20), where learning and testing data were chosen randomly. The obtained results were compared to the measurements and to those calculated using an equivalent MLR/GA model presented in [23].

The parameters of the GA for both hybrids were as follows: the number of individuals in the population pool (the number of ANN or MLR models) was 70, while the number of generations was 50. Binary tournament selection accompanied with a single point crossover and uniform mutation with a possibility of 0.9 and 0.125 were used in the analysis.

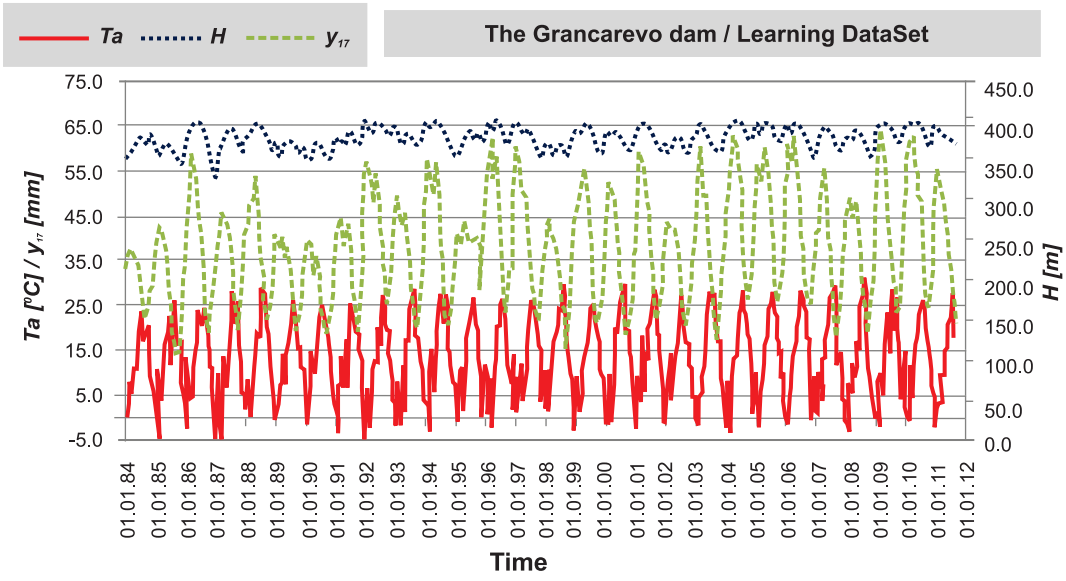

Fig. 10. A learning dataset for Grancarevo dam from the period 1984–2011.

All the tests were performed on Intel i3-3120 M CPU @2.50 GHz with 4 GB of memory.

4.2.1. The ANN/GA hybrid parameters and results

The structure of the chromosome used for modeling radial displacement of point P1 is formed according to Table 1.

Fig. 11 shows an example of a chromosome that represents an ANN individual with 11 hidden layers, each with 20 neurons, an RPROP learning algorithm and TANH activation function. The LP1 and LP2 parameters have values $\eta^+ = 1.2$ and $\eta^- = 0.5$.

Initial values of weighting coefficients $\mathbf{W_0}$ were chosen randomly from the range [−0.5,0.5]. In order to reduce the influence of the stochastic choice of $\mathbf{W_0}$, the learning process for each ANN individual was repeated five times ($R = 5$).

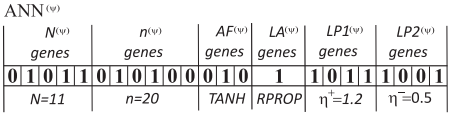

Fig. 11. An example of a chromosome representing an ANN individual.

The modeling of point P1 radial displacement was performed for each of the LDS-TDS pairs and different predictor sets, and the results of the analyses are given in Table 2.

Table 1
Structure of binary ANN chromosome.

Genes	Denotation	Note	Range	Number of bits	Encoding example
Number of hidden layers	$N^{(\psi)}$		[1–32]	5	00010 2 hidden layers
Number of neurons in hidden layers	$n^{(\psi)}$		[1–64]	6	000111 7 neurons per hidden layer
Activation function	$AF^{(\psi)}$		[0–7]	3	001-*Sin AF* 010- *Hyperbolic tangent AF* …
Learning algorithm	$LA^{(\psi)}$			1	0-BPGD algorithm 1-RPROP algorithm
Learning rule	$LP1^{(\psi)}$	for RPROP: η^+	[1.0–1.60]	4	0001 = 1.04; … ; 0101 = 1.20; 1001 = 1.36;…
	$LP1^{(\psi)}$	for BPGD: η	[0.2–0.95]	4	0001 = 0.25; … ; 1110 = 0.90; 1111 = 0.95
	$LP2^{(\psi)}$	for RPROP: η^-	[0–1]	4	0001 = 0.15; … ; 1001 = 0.50; … ; 1110 = 0.90; ….
	$LP2^{(\psi)}$	for BPGD: μ	[0.4–0.9]	4	…, 1101 = 0.83; 1110 = 0.86; 1111 = 0.90;…

Table 2
Quality of ANN/GA models expressed by $RMSE_{Train}$ and $RMSE_{Test}$.

ANN/GA	$RMSE_{Train}$ [mm]				$RMSE_{Test}$ [mm]			
Cross valid.	predictors				predictors			
LDS:TDS size ratio	H, T_a t	H, T_a t, d	H, T_a $t, d20$	H, T_a $t, d50$	H, T_a t	H, T_a t, d	H, T_a $t, d20$	H, T_a $t, d50$
60–40 (3031:2021)	4.3701	2.1384	1.8815	1.5252	4.6598	2.1044	1.8854	1.5845
70–30 (3536:1516)	4.3745	2.1492	2.3184	3.2118	4.3759	2.1044	2.2654	3.2673
75–25 (3789:1263)	4.2113	2.0638	1.8562	5.8963	4.5877	2.0655	1.8647	5.8434
80–20 (4042:1010)	4.5001	1.8538	1.7574	1.6811	4.8433	2.3182	1.9145	1.6739

Table 3
Regressors used for modeling radial displacement of point P1 (y_{17}).

ρ^*	Regressors			
	Water level		Temperature	Aging
ρ_I	$H,\ H^2, H^3$ $\sqrt{H}$	$T_a,\ T_a^2,\ T_a^3,\ T_a^4,\ T_a^5$ $e^{-T_a}, e^{-T_a/2}$		t, t^2 $\sqrt{t}$ $e^{-t},\ e^{-t/2},\ e^{-t/4}$
ρ_{II}	$H,\ H^2, H^3$ $\sqrt{H}$	$T_a,\ T_a^2,\ T_a^3,\ T_a^4,\ T_a^5$ $e^{-T_a}, e^{-T_a/2}$	$\sin(2k\cdot\pi\cdot d/365),\ k=1,2,4$ $\cos(2k\cdot\pi\cdot d/365),\ k=1,2,4$	t, t^2 $\sqrt{t}$ $e^{-t},\ e^{-t/2},\ e^{-t/4}$

* Pool of regressors

Table 4
Quality of the MLR models expressed by $RMSE_{Test}$.

MLR/GA	$RMSE_{Train}$ [mm]				$RMSE_{Test}$ [mm]			
CrossValidation	*predictors*				*predictors*			
LDS:TDS size ratio	H, T_a t	H, T_a t, d	H, T_a $t, d20$	H, T_a $t, d50$	H, T_a t	H, T_a t, d	H, T_a $t, d20$	H, T_a $t, d50$
60–40 (3031:2021)	5.0362	2.0147	2.0125	2.0138	5.0891	2.0144	2.0120	2.0134
70–30 (3536:1516)	5.1048	2.1378	2.1375	2.1374	4.9386	2.0750	2.0735	2.0748
75–25 (3789:1263)	5.0420	2.1504	2.1524	2.1539	5.0947	2.0170	2.0193	2.0237
80–20 (4042:1010)	5.0556	2.0514	2.0520	2.0513	5.1230	2.4311	2.4339	2.4370

Due to the complexity of neural networks the average duration of model generation varied from 1.5 to 5 h. Models with input variables H, T_a and t showed stable behavior for all predefined LDS:TDS ratios. The $RMSE_{Test}$ for these cases was approximately 4.5 mm (9% of the point P1 displacement range).

Including variable d (day of the year), the quality of the model was significantly improved, so the $RMSE_{Test}$ for predictor set H, T_a, t and d had a value of approximately 2 mm, which is about 4% of the point P1 displacement range. Nevertheless, in tests with *dummy* predictors $d20$ and $d50$, a certain instability of model quality for various LDS:TDS ratios was noticed.

In more than 90% of tests, neural networks with the best characteristics have used the *hyperbolic tangent* activation function (TANH). The most accurate ANN model ($RMSE_{Test} = 2.0655$ mm) was obtained for predictors H, Ta, t and d with 75:25 LDS:TDS ratio. The optimized neural network had 5 hidden layers, each with 30 neurons. The activation function was TANH, while the learning algorithm was RPROP with $\eta^+ = 1.15$ and $\eta^- = 0.65$. The optimization process took about 135 min.

4.2.2. The MLR/GA hybrid parameters and results

With the aim of making the ANN/GA and MLR/GA dam models comparable, the complexity of multiple linear regression models was not constrained (complexity parameter in the MLR/GA hybrid [23] was set to $\lambda = 0$). We have created two pools of possible regressors (ρ_I and ρ_{II}) that describe the radial displacements of point P1, contributed by hydrostatic pressure (H), temperature variation (T_a), and ageing (t). Similarly to ANN models, variable d was introduced for modeling the thermal effects represented by trigonometric functions as temperature cyclic (seasonal) loadings (Table 3.).

In the pool ρ_I we considered the set of 17 potential regressors containing all of the available variables without trigonometric forms. The pool ρ_{II} had 25 regressors which covered all predictors in various forms, including the trigonometric form of input variable d. In order to test the robustness to temperature phase offset, we performed additional analyses where input variable d was replaced by dummy variables $d20$ and $d50$.

The results obtained using the MLR/GA hybrid for cross-validation tests are given in Table 4.

In the case when there were no trigonometric regressors in the pool (ρ_I set of regressors), errors of the models were approximately 5 mm (10% of the point P1 displacement range). After including variable d and trigonometric regressors, the error $RMSE_{Test}$ was reduced to approximately 2 mm, which is about 4% of the point P1 displacement range. Usage of dummy variables $d20$ and $d50$ instead of variable d had no significant influence on the error.

The error $RMSE_{Test}$ for all cross-validation tests was approximately 2 mm, except for LDS:TDS = 80:20, where the $RMSE_{Test}$ was a bit higher, about 2.43 mm.

The best of all optimized models had the error $RMSE_{Test} = 2.0144$ mm, which was obtained for LDS:TDS = 60:40, the set of predictors $\{H, T_a, t, d\}$ and 22 regressors from the pool ρ_{II} (Eq. 7):

$$\begin{aligned} y_{17} = {} & (7.906\text{e}+05) + (5.974\text{e}+03)\cdot H \\ & - (5.482\text{e}+00)\cdot H^2 + (0.020\text{e}-03)\cdot H^3 \\ & - (1.215\text{e}+05)\cdot\sqrt{H} - (1.591\text{e}-01)\cdot T_a \\ & - (1.682\text{e}-02)\cdot T_a^2 + (2.623\text{e}-03)\cdot T_a^3 \\ & - (1.360\text{e}-04)\cdot T_a^4 + (2.148\text{e}-06)\cdot T_a^5 \\ & + (3.656\text{e}-03)\cdot(e^{-T_a}) \\ & + (8.787\text{e}+00)\cdot\sin(2\cdot\pi\cdot d/365) \\ & + (8.795\text{e}-02)\cdot\sin(4\cdot\pi\cdot d/365) \\ & + (4.055\text{e}+00)\cdot\cos(2\cdot\pi\cdot d/365) \\ & + (8.806\text{e}-01)\cdot\cos(4\cdot\pi\cdot d/365) \\ & + (8.543\text{e}-02)\cdot\cos(6\cdot\pi\cdot d/365) \\ & + (1.186\text{e}-01)\cdot\cos(8\cdot\pi\cdot d/365) \\ & + (6.263\text{e}+03)\cdot t - (5.263\text{e}+01)\cdot t^2 \\ & - (3.819\text{e}+04)\cdot\sqrt{t} \\ & - (5.435\text{e}+04)\cdot e^{-t} + (1.334\text{e}+04)\cdot e^{-t/2} \\ & - (4.239\text{e}+04)\cdot e^{-t/4} \end{aligned} \quad (7)$$

The optimization of MLR models took 5 to 10 min.

4.3. Discussion

Fig. 12 shows the aggregated results of the performed tests of the ANN/GA and MLR/GA hybrids, where the quality of the models is expressed by the $RMSE_{Test}$.

It can be seen from the Figure that, in the analyses where H, T_a and t were used as input variables, the ANN/GA hybrid had shown

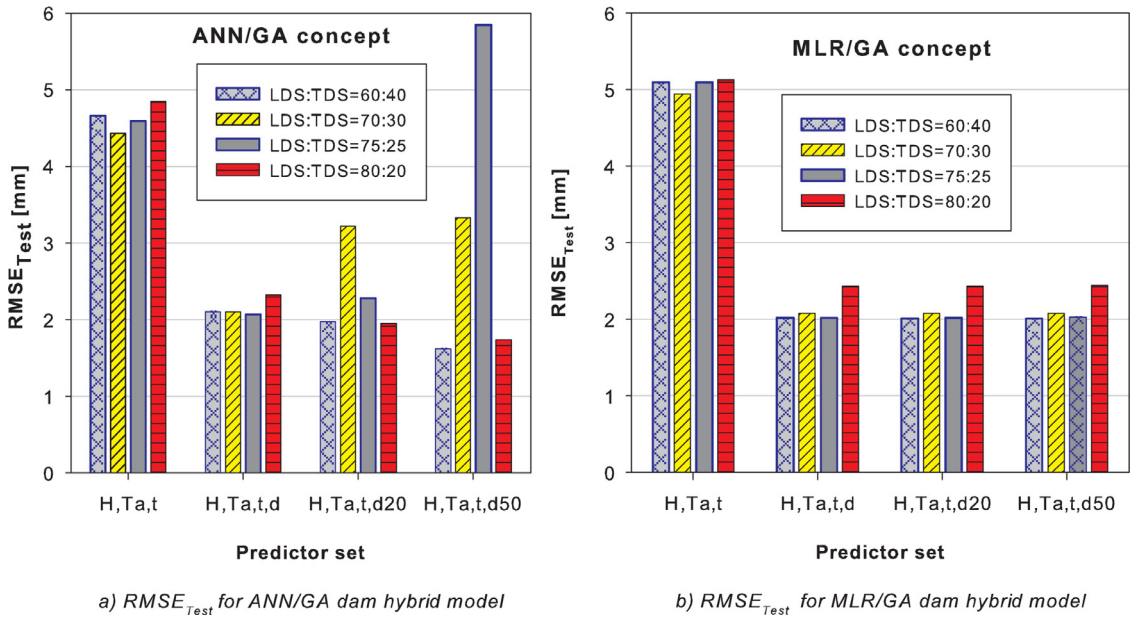

Fig. 12. $RMSE_{Test}$ values for different sets of predictors and LDS:TDS ratios in ANN and MLR models.

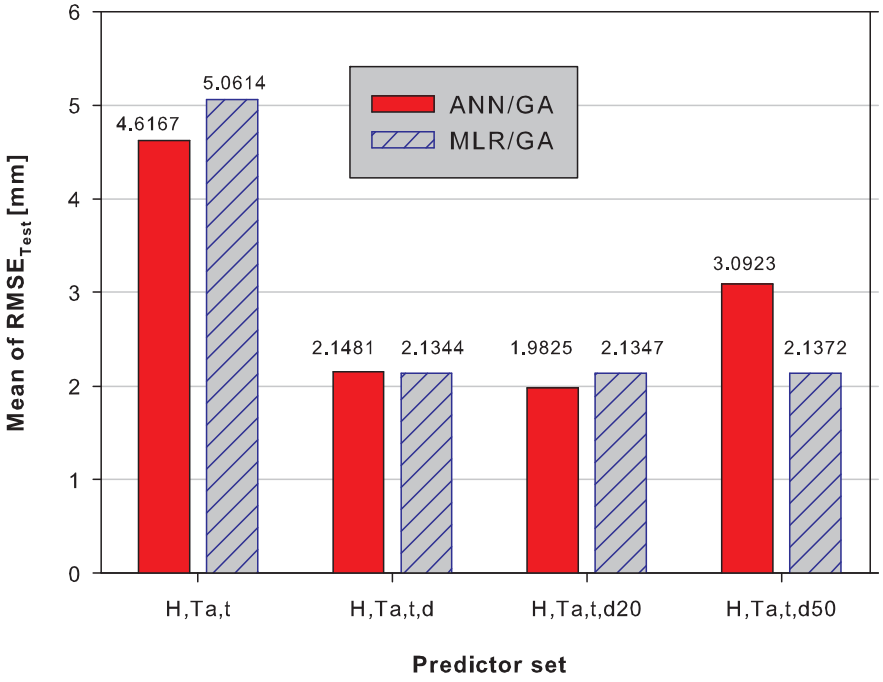

Fig. 13. Average values of $RMSE_{Test}$ for different subsets of predictors.

better accuracy than MLR/GA. The $RMSE_{Test}$ of the ANN models was approximately 4.5 mm, which is about 10% lower compared to the $RMSE_{Test}$ of the MLR/GA hybrid, which was approximately 5 mm.

The quality of both the ANN/GA and MLR/GA hybrids was significantly increased when input variable *d* was introduced, so the $RMSE_{Test}$ was reduced to a value of approximately 2 mm. The improvement of the quality of the ANN and MLR models that use day of the year as an input variable can be explained by the capacity of variable *d* to better represent the temperature cyclic (seasonal) loadings.

The tests in which variable *d* was replaced by dummy variables *d*20 and *d*50 showed that ANN models were unstable when temperature offset was present. The instability of ANN models is particularly noticeable when sets of predictors contained variable *d*50 (Fig. 12a), where $RMSE_{Test}$ values varied from 1.6 mm to 5.84 mm. In contrast to the ANN models, MLR models were shown as very stable in tests with dummy variables *d*20 and *d*50 (Fig. 12b). The stability of MLR models can be explained by the capacity of trigonometric regressors to model phase offset by simple superposition of sine and cosine functions.

Fig. 13 shows the average values of $RMSE_{Test}$ errors of the ANN/GA and MLR/GA models for each of the predictor subsets. It can be seen from the figure that the average error in tests with dummy variable *d*50 is also significantly worse when the ANN/GA is used.

Regarding computational costs, the average time needed to generate an optimized ANN model is dramatically higher than the time spent to generate an equivalent MLR model (1.5 to 5 h for the ANN/GA compared to 5–10 min for the MLR/GA). The time to reach

a trained model can further increase with the expansion of learning dataset. However, in dam modeling we usually deal with daily measurements during few decades, which is up to 10 thousands data samples. In this study we have used a learning dataset of about 5 thousands data samples, so the presented computational times can be considered as an average or near-worst scenario. On the other hand, although the maximal number of hidden layers and the maximal number of nodes per layer were set to 32 and 64 (see Table 1), the most of optimized models have consisted of 5–7 hidden layers with 20–30 neurons. Therefore, the computational times could be significantly reduced by narrowing GA constraints to reasonable values (for example, maximum 8 hidden layers and maximum 32 nodes per layer).

5. Conclusion

In this paper, DEVONNA, the self-tuning software system for dam behavior modeling based on an evolving artificial neural network is presented. In order to deal with the variant set of predictor variables and the permanently increasing database of measurements in real-world problems, we developed a methodology for the generation of an ANN dam model that is optimized for given conditions using genetic algorithms. In contrast to the existing solutions, permanent variability of inputs and learning datasets require a model where most of the ANN parameters, such as the number of hidden layers, number of neurons per layer, activation function, learning algorithm and learning parameters are optimized. According to the corresponding mathematical programming model, we developed an automated system based on genetic algorithms that adapts an ANN network to fit currently active sensors and is learned using the latest historical data.

The developed ANN/GA hybrid was validated using the Grancarevo dam case study (at the Trebisnjica river located in the Republic of Srpska), in which the radial displacements of a point inside the dam structure as a function of the headwater, temperature and time have been modeled. In order to test performances of the ANN/GA hybrid, the results were compared to measurements and to those obtained by an equivalent MLR/GA hybrid.

The tests performed have shown that the developed ANN/GA hybrid can give prediction of structural dam behavior with approximately 10% better accuracy compared to models based on the MLR/GA concept. However, in contrast to MLR models, which are resistant to the temperature phase offset present at different geographical locations, the ANN models showed very unstable behavior under such conditions. In order to keep the accuracy advantage over MLR models, ANN models should be manually tuned if day of the year is used as an input parameter. In addition, generating an optimized ANN dam model can last for several hours, which is significantly longer than the 5 to 10 min needed to generate an equivalent MLR model. This drawback of the ANN/GA hybrid can be overcome by using parallel frameworks for GA optimization such as WoBinGO [45], which provides almost linear speed-up of evolutionary algorithms. Using WoBinGO on an average Grid infrastructure consisting of hundred nodes would shorten computational time of the ANN/GA hybrid to a few minutes.

The main limitation of this approach lies in the fact that it does not directly consider mechanical properties and any possible damage. Additional analysis in the form of non-destructive tests (static and dynamical), computational mechanical modeling and inverse analysis will also be required.

Acknowledgments

Part of this research was supported by the Ministry of Science in Serbia, Grants III41007, OI174028 and TR37013, and FP7 ICT-2007-2-5.3 (224297) ARTreat project.

References

[1] De Sortis A, Paoliani P. Statistical analysis and structural identification in concrete dam monitoring. Eng Struct 2007;29:110–20.
[2] Arrdito R, Cocchetti G. Statistical approach to damage diagnostic of concrete dam by radar monitoring: formulation and pseudo-experimental test. Eng Struct 2006;28:2036–45.
[3] Hu D, Zhou Z, Li Y, Wu X. Dam safety analysis based on stepwise regression model. Adv Mater Res 2011;204-210:2158–61.
[4] Szostak-Chrzanowski A, Chrzanowski A, Massiera M. Use of deformation monitoring results in solving geomechanical problems, case studies. Eng Geology 2005;79(1-2):3–12.
[5] Yifeng C, Ran H, Wenbo L, Dianqing L, Chuangbing Z. Modeling coupled processes of non-steady seepage flow and non-linear deformation for a concrete–faced rockfill dam. Comput Struct 2011;89(13–14):1333–51.
[6] Rankovic V, Grujovic N, Divac D, Milivojevic N, Novakovic A. Modeling of dam behavior based on neuro fuzzy identification. Eng Struct 2012;35:107–13.
[7] Hagan MT, Demuth HB, Beale MH, De Jesús O. Neural Network Design. Boston, U.S.A.: PWS Publishing Company; 1996.
[8] Mata J. Interpretation of concrete dam behavior with artificial neural network and multiple linear regression models. Eng Struct 2011;33:903–10.
[9] Gholizadeh S, Samavati OA. Structural optimization by wavelet transforms and neural networks. Appl Math Model 2011;35:915–29.
[10] Gholizadeh S, Seyedpoor SM. Shape optimization of arch dam by metaheuristics and neural networks for frequency constraints. Scientia Iranica A: Civil Eng 2011;18(5):1020–7.
[11] Chang X, Dongjie Y, Chengfa D. Hybrid GA/SIMPLS as alternative regression model in dam deformation analysis. Eng Appl Artif Intell 2012;25:468–75.
[12] Salajegheh E, Gholizadeh E. Optimum design of structures by an improved genetic algorithm using neural networks. Adv Eng Softw 2005;36:757–67.
[13] Zuo W, Xu T, Zhang H, Xu T. Fast structural optimization with frequency constraints by genetic algorithm using adaptive eigenvalue reanalysis methods. Struct Multidisciplinary Opt 2011;43:799–810.
[14] Gomes HM. Truss optimization with dynamic constraints using a particle swarm algorithm. Expert Syst Appl 2011;38:957–68.
[15] Xi GY, Yue JP, Zhou BX, Tang P. Application of an artificial immune algorithm on statistical model of dam displacement. Comput Math Appl 2011;62:3980–6.
[16] Yildiz AR. A new design optimization framework based on immune algorithm and Taguchi's method. Comput Indus 2009;60:613–20.
[17] Majdi M, Beiki M. Evolving neural network using a genetic algorithm for predicting the deformation modulus of rock masses. Int J Rock Mech Mining Sci 2010;47:246–53.
[18] Karimi I, Khaji N, Ahmadi MT, Mirzayee M. Evolving neural network using a genetic algorithm for predicting the deformation modulus of rock masses. Eng Struct 2010;32:3583–91.
[19] Zhou C-B, Liua W, Chen Y-F, Hua R, Wei K. Inverse modeling of leakage through a rockfill dam foundation during its construction stage using transient flow model, neural network and genetic algorithm. Eng Geol 2015;187:183–95.
[20] Fedele R, Maier G, Miller B. Evolving neural network using a genetic algorithm for predicting the deformation modulus of rock masses. Int J Fracture 2006;137:151–72.
[21] Hooshyaripor F, Tahershamsi A, Golian S. Application of copula method and neural networks for predicting peak outflow from breached embankments. J Hydro-environ Res 2014;8:292–303.
[22] Hecht-Nielsen R. Kolmogorov's mapping neural network existence theorem. In: Proceedings of first IEEE international conference on neural networks, vol. 3; 1987. p. 11–14.
[23] Stojanovic B, Milivojevic M, Ivanovic M, Milivojevic N, Divac D. Adaptive system for dam behavior modeling based on linear regression and genetic algorithms. Adv Eng Softw 2013;65:182–90.
[24] Yu Z. Feed-forward Neural Networks and Their Applications in Forecasting. University of Houston; 2000.
[25] Heaton J. Programming Neural Networks with Encog3 in C#. Heaton Research, Inc.; 2015. <http://goo.gl/Zo8k70> (accessed 3.6.2015).
[26] Riedmiller M, Braun H. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. Neural networks. In: Proceedings of IEEE International Conference on: San Francisco, CA, vol.1; 1993. p. 586–91.
[27] Riedmiller M. Advanced supervised learning in multi-layer perceptrons – from backpropagation to adaptive learning algorithms. Comput. Stand. Interfaces 1994;16:265–78.
[28] Anastasiadis AD, Magoulas GD, Vrahatis MN. New globally convergent training scheme based on the resilient propagation algorithm. Neurocomputing 2005;64:253–70.
[29] Günther F, Fritsch S. Neuralnet: training of neural networks. R J 2010;2/1:30–8.
[30] Rumelhart DE, Mc Clelland JL. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations. Cambridge MA: MIT Press; 1986.
[31] Topping BHV, Sziveri J, Bahreinejad A, Leite JPB, Cheng B. Parallel processing, neural networks and genetic algorithms. Adv Eng Softw 1998;29(10):763–86.
[32] Poulton MM. Neural networks as an intelligence amplification tool: a review of applications. J Geophys 2002;67(3):979–93.
[33] Swingler K. Applying neural networks: a practical guide. NewYork: Academic Press; 1996.
[34] Looney CG. Advances in feed-forward neural networks: demystifying knowledge acquiring black boxes. IEEE Trans Knowl Data Eng 1996;8(2):211–26.

[35] Nelson M, Illingworth WT. A practical guide to neural nets. Reading. Mass: Addison-Wesley; 1990.
[36] Looney CG. Pattern recognition using neural networks, theory and algorithms for engineers and scientists. NewYork: Oxford University Press; 1997.
[37] Messer K, Kittler J. Choosing an optimal neural network size to aid search through a large image database. In: Proceedings of ninth British machine vision conference (BMVC98), University of Southampton; 1998. p. 235–44.
[38] Holland JH. Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press; 1975.
[39] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Boston: Addison-Wesley Longman Publishing Co.; 1989.
[40] Anagnostopoulos KP, Mamanis G. A portfolio optimization model with three objectives and discrete variables. Comput Oper Res 2010;37:1285–97.
[41] Brands T, Van Berkum EC. Performance of a genetic algorithm for solving the multi-objective, multimodal transportation network design problem. Int J Transport 2014;2(1):1–20.
[42] Haddad K, Rahman A, Zaman MA, Shrestha S. Applicability of Monte Carlo cross validation technique for model development and validation using generalised least squares regression. J Hydrol 2013;482:119–28.
[43] Zhang Y, Yin Y, Guo D, Yu X, Xiao L. Cross-validation based weights and structure determination of Chebyshev-polynomial neural networks for pattern classification. Patt. Recog. 2014;47:3414–28.
[44] Bergmeir C, Benítez JM. On the use of cross-validation for time series predictor evaluation. Inform. Sci. 2012;191:192–213.
[45] Ivanovic M, Simic V, Stojanovic B, Kaplarevic-Malisic A, Marovic B. Elastic grid resource provisioning with WoBinGO: a parallel framework for genetic algorithm based optimization. Future Generation Comput. Syst. 2015;42:44–54.